
Acton Documentation

Release 0.3.3

Matthew Alger & Cheng Soon Ong

Feb 15, 2017

1	acton	3
1.1	acton package	3
2	Developer Documentation	7
2.1	Contributing	7
2.2	Adding a New Predictor	7
2.3	Adding a New Recommender	8
3	Indices and tables	9
	Python Module Index	11

Contents:

1.1 acton package

1.1.1 Subpackages

acton.proto package

Submodules

acton.proto.io module

Functions for reading/writing to protobufs.

`acton.proto.io.get_ndarray` (*data:* list, *shape:* tuple, *dtype:* str) → <MagicMock id='139892016796504'>

Converts a list of values into an array.

Parameters

- **data** – Raw array data.
- **shape** – Shape of the resulting array.
- **dtype** – Data type of the resulting array.

Returns Array with the given data, shape, and dtype.

Return type numpy.ndarray

`acton.proto.io.read_metadata` (*file:* typing.Union[str, typing.BinaryIO]) → bytes

Reads metadata from a protobufs file.

Parameters **file** – Path to binary file, or file itself.

Returns Metadata.

Return type bytes

`acton.proto.io.read_proto` ()

Reads a protobuf from a .proto file.

Parameters

- **path** – Path to the .proto file.
- **Proto** – Protocol message class (from the generated protobuf module).

Returns The parsed protobuf.

Return type `GeneratedProtocolMessageType`

```
acton.proto.io.read_protos (file: typing.Union[str, typing.BinaryIO], Proto: <MagicMock
                             id='139892016817264'>) → <MagicMock name='mock()'
                             id='139892016816368'>
```

Reads many protobufs from a file.

Parameters

- **file** – Path to binary file, or file itself.
- **Proto** – Protocol message class (from the generated protobuf module).

Yields `GeneratedProtocolMessageType` – A parsed protobuf.

```
acton.proto.io.write_proto()
```

Serialises a protobuf to a file.

Parameters

- **path** – Path to binary file. Will be overwritten.
- **proto** – Protobuf to write to file.

```
acton.proto.io.write_protos (path: str, metadata: bytes = b'')
```

Serialises many protobufs to a file.

Parameters

- **path** – Path to binary file. Will be overwritten.
- **metadata** – Optional bytestring to prepend to the file.

Notes

Coroutine. Accepts protobufs, or None to terminate and close file.

acton.proto.predictors_pb2 module

acton.proto.wrappers module

Module contents

1.1.2 Submodules

1.1.3 acton.acton module

1.1.4 acton.cli module

1.1.5 acton.database module

1.1.6 acton.kde_predictor module

A predictor that uses KDE to classify instances.

class `acton.kde_predictor.KDEClassifier` (*bandwidth=1.0*)

Bases: `BaseEstimator`, `ClassifierMixin`

A classifier using kernel density estimation to classify instances.

fit (*X*, *y*)

Fits kernel density models to the data.

Parameters

- **X** (*array_like*, *shape* (*n_samples*, *n_features*)) – List of *n_features*-dimensional data points. Each row corresponds to a single data point.
- **y** (*array-like*, *shape* (*n_samples*,)) – Target vector relative to X.

predict (*X*)

Predicts class labels.

Parameters X (*array_like*, *shape* (*n_samples*, *n_features*)) – List of *n_features*-dimensional data points. Each row corresponds to a single data point.

predict_proba (*X*)

Predicts class probabilities.

Class probabilities are normalised log densities of the kernel density estimates.

Parameters X (*array_like*, *shape* (*n_samples*, *n_features*)) – List of *n_features*-dimensional data points. Each row corresponds to a single data point.

1.1.7 acton.labellers module

1.1.8 acton.plot module

1.1.9 acton.predictors module

1.1.10 acton.recommenders module

1.1.11 Module contents

Developer Documentation

2.1 Contributing

We accept pull requests on GitHub. Contributions must be PEP8 compliant and pass formatting and function tests in the test script `/test`.

2.2 Adding a New Predictor

A predictor is a class that implements `acton.predictors.Predictor`. Adding a new predictor amounts to implementing a subclass of `Predictor` and registering it in `acton.predictors.PREDICTORS`.

Predictors must implement:

- `__init__(db: acton.database.Database, *args, **kwargs)`, which stores a reference to the database (and does any other initialisation).
- `fit(ids: Iterable[int])`, which takes an iterable of IDs and fits a model to the associated features and labels,
- `predict(ids: Sequence[int]) -> numpy.ndarray`, which takes a sequence of IDs and predicts the associated labels.
- `reference_predict(ids: Sequence[int]) -> numpy.ndarray`, which behaves the same as `predict` but uses the best possible model.

Predictors should store data-based values such as the model in attributes ending in an underscore, e.g. `self.model_`.

2.2.1 Why Does Acton Use Predictor?

Acton makes use of `Predictor` classes, which are often just wrappers for scikit-learn classes. This raises the question: Why not just use scikit-learn classes?

This design decision was made because Acton must support predictors that do not fit the scikit-learn API, and so using scikit-learn predictors directly would mean that there is no unified API for predictors. An example of where Acton diverges from scikit-learn is that scikit-learn does not support multiple labellers.

2.3 Adding a New Recommender

A recommender is a class that implements `acton.recommenders.Recommender`. Adding a new recommender amounts to implementing a subclass of `Recommender` and registering it in `acton.recommenders.RECOMMENDERS`.

Recommenders must implement:

- `__init__(db: acton.database.Database, *args, **kwargs)`, which stores a reference to the database (and does any other initialisation).
- `recommend(ids: Iterable[int], predictions: numpy.ndarray, n: int=1, diversity: float=0.5) -> Sequence[int]`, which recommends `n` IDs from the given IDs based on the associated predictions.

Indices and tables

- `genindex`
- `modindex`
- `search`

a

acton, [5](#)
acton.kde_predictor, [4](#)
acton.proto, [4](#)
acton.proto.io, [3](#)

A

[acton \(module\)](#), 5
[acton.kde_predictor \(module\)](#), 4
[acton.proto \(module\)](#), 4
[acton.proto.io \(module\)](#), 3

F

[fit\(\) \(acton.kde_predictor.KDEClassifier method\)](#), 5

G

[get_ndarray\(\) \(in module acton.proto.io\)](#), 3

K

[KDEClassifier \(class in acton.kde_predictor\)](#), 4

P

[predict\(\) \(acton.kde_predictor.KDEClassifier method\)](#), 5
[predict_proba\(\) \(acton.kde_predictor.KDEClassifier method\)](#), 5

R

[read_metadata\(\) \(in module acton.proto.io\)](#), 3
[read_proto\(\) \(in module acton.proto.io\)](#), 3
[read_protos\(\) \(in module acton.proto.io\)](#), 4

W

[write_proto\(\) \(in module acton.proto.io\)](#), 4
[write_protos\(\) \(in module acton.proto.io\)](#), 4